Applying Ant Colony Optimization (ACO) - Traveling Sales-person Problem (TSP): Program Result Overviews

by Kirill Ivanov

Table Of Contents

1 Introduction

The first application of an ant colony optimization algorithm was done using the traveling salesman problem (TSP) as a test problem. The main reasons why the TSP, one of the most studied NP-hard[5] problems in combinatorial optimization, was chosen are that it is a shortest path problem to which the ant colony metaphor is easily adapted and that it is a didactic problem (that is, it is very easy to understand and explanations of the algorithm behavior are not obscured by too many technicalities).

Ant algorithms were inspired by the observation of real ant colonies [1]. Ants are social insects, that is, insects that live in colonies and whose behavior is directed more to the survival of the colony as a whole than to that of a single individual component of the colony. Social insects have captured the attention of many scientists because of the high structuration level their colonies can achieve, especially when compared to the relative simplicity of the colony's individuals. An important and interesting behavior of ant colonies is their foraging behavior, and, in particular, how ants can find shortest paths between food sources and their nest.

While walking from food sources to the nest and vice versa, ants deposit on the ground a substance called pheromone, forming in this way a pheromone trail. Ants can smell pheromone and, when choosing their way, they tend to choose, in probability, paths marked by strong pheromone concentrations. The pheromone trail allows the ants to find their way back to the food source (or to the nest). Also, it can be used by other ants to find the location of the food sources found by their nestmates [1].

2 Why Ant System (AS)

Ant System (AS) was the first ACO algorithm [2, 3]. Its importance resides mainly in being the prototype of a number of ant algorithms which have found many interesting and successful applications.

In AS artificial ants build solutions (tours) of the TSP by moving on the problem graph from one city to another. The algorithm executes t_{max} iterations, in the following indexed by t. During each iteration m ants build a tour executing n steps in which a probabilistic decision (state transition) rule is applied. In practice, when in node i the ant chooses the node j to move to, and the arc (i, j) is added to the tour under construction. This step is repeated until the ant has completed its tour.

Three AS algorithms have been defined, which differ by the way pheromone trails are updated [2, 3, 4]. These algorithms are called ***ant-density***, ***ant-quantity***, and ***ant-cycle***. In ant-density and ant-quantity ants deposit pheromone while building a solution, while in ant-cycle ants deposit pheromone after they have built a complete tour.

Preliminary experiments run on a set of benchmark problems have shown that ant-cycle's performance was much better than that of the other two algorithms. Consequently, research on AS was directed towards a better understanding of the characteristics of ant-cycle, which is now known as Ant System, while the other two algorithms were abandoned.

As it was mentioned above, in AS after ants have built their tours, each ant deposits pheromone on pheromone trail variables associated to the visited arcs to make the visited arcs become more desirable for future ants (that is, online delayed pheromone update is at work). Then the ants die. In AS no daemon activities are performed, while the pheromone evaporation procedure, which happens just before ants start to deposit pheromone, is interleaved with the ants activity.

The amount of pheromone trail τ_{ij} associated to arc(i,j) is intended to represent the learned desirability of choosing city j when in city i (which also corresponds to the desirability that the arc(i,j) belong to the tour built by an ant). The pheromone trail information is changed during problem solution to reflect the experience acquired by ants during problem solving. Ants deposit an amount of pheromone proportional to the quality of the solutions they produced: the shorter the tour generated by an ant, the greater the amount of pheromone it deposits on the arcs which it used to generate the tour. This choice helps to direct search towards good solutions. The main role of pheromone evaporation is to avoid stagnation, that is, the situation in which all ants end up doing the same tour.

The memory (or internal state) of each ant contains the already visited cities and is called tabu list (in the following it will be continued to use the term tabu list to indicate the ant's memory). The memory is used to define, for each ant k, the set of cities that an ant located on city i still has to visit. By exploiting the memory therefore an ant k can build feasible solutions by an implicit state-space graph generation (in the TSP this corresponds to visiting a city exactly once). Also, memory allows the ant to cover the same path to deposit online delayed pheromone on the visited arcs.

A general definition of the traveling salesman [1] problem is the following. Consider a set N of nodes, representing cities, and a set E of arcs fully connecting the nodes N. Let dij be the length of the arc (i, j) C E, that is the distance between cities i and j, with i, j C N. The TSP is the problem of finding a minimal length Hamiltonian circuit on the graph G = (N,E), where an Hamiltonian circuit of graph G is a closed tour visiting once and only once all the n = |N| nodes of G, and its length is given by the sum of the lengths of all the arcs of which it is composed.

3 Program Algorithm (An Ant System for TSP)

Ant Sysem (AS) is a fairly early version of currently called Ant Colony Optimization (ACO) algorithm. Here in order to learn how do artificial ants behave, it applies AS to TSP [1].

Assume all the cordinate of each of the N cities which also allow us to calculate d_{ij} — the distance from city i to j.

1. Create p ants and each of these p ants is put on a city chosen at random.

2. Each ant starting with the initially chosen city moves to the closest city iteratively (ties are broke at random). This is called Nearest-neighbor Procedure.

3. Initialize Pheromon Matrix [τ_{ij}]. All the τ_{ij} is initialized to the identical value of

$$\tau_{ij}^0 = \frac{Number\ of\ Ants}{sum\ of\ the\ lengths\ of\ the\ tour\ obtained\ by\ the\ above\ Nearest\text{-}Neighbour}$$

4. After completed their tour, each ant deposits pheromon on the paths the ant have followed

5. Calculate η_{ij} — heuristic desirability from city i to j which is usually set to $1/d_{ij}$.

6. Construct a tour for each ant such that k-th ant at city i choose a city j with the probability

$$p_{ij}^k = \frac{\tau_{ij}^\alpha \cdot \eta_{ij}^\beta}{\sum_{l}^{k-1} \tau_{il}^\alpha \cdot \eta_{il}^\beta}$$

where α is usually set to 1 and β is chosen from 2 to 5.

7. Update the pheromon

(1) **Evaporate Pheromon** Evaporate the previously put pheromon as

$$\tau_{ij} = (1-\rho)\tau_{ij}$$

(2) **Deposit Pheromon** Deposit pheromon as

$$\tau_{ij} = \tau_{ij} + \sum_{k=1}^{p} \delta\tau_{ij}^k$$

where $\delta\tau_{kij} = 1/C_k$ if the arc(i,j) belongs to the tour, otherwise 0. C_k is the length of the tour.

4 Program result analysis

The main idea of this program is creating the probabilistic model of choosing a way to make a tour. So every new node (a way from city *i* to *j*) is choosed with the probability

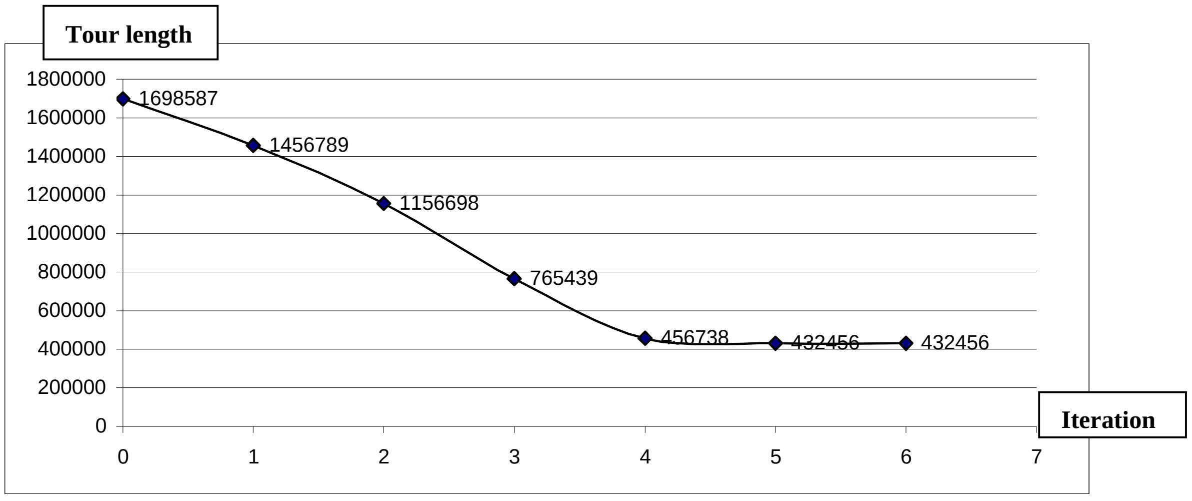
$$p_{ij}^{k} = \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{l}^{k-1} \tau_{il}^{\alpha} \cdot \eta_{il}^{\beta}}$$

This means that it is choosed node *j* from node *i* as max multiplication value $\tau_{ij}*\eta_{ij}$ whrere η_{ij} - heuristic desirability from city i to j which is usually set to $1/d_{ij}$ and τ_{ij} - the amount of pheromone trail associated to arc(i,j) is intended to represent the learned desirability of choosing city j when in city i (which also corresponds to the desirability that the arc(i,j) belong to the tour built by an ant).

In the program model, the probability of choosing a branch at a certain time depends on the total amount of pheromone on the branch, which in turn is proportional to the number of ants that used the branch until that time.

Depositing amount of pheromone is a function of the quality of the solution found.

After every iteration (all ants have built their solutions) the program finds the best one (the shortest tour of all completed tours by the all ants). Then if makes some advanced iterations to make sure that there is no more best solution. With another words the programs continues to perform iterations until last several iterations gives the same solution. All this is clear to see on the graph (iterations; a tour length).



- Graph of program iterations result for finding the minimal tour -

5 Conclusions

In this paper it has introduced (and applied in the program) the ant colony optimization (ACO) meta-heuristic and it has been given an overview of ACO algorithms. ACO is a novel and very promising research field situated at the crossing between artificial life and operations research. The ant colony optimization meta-heuristic belongs to the relatively new wave of stochastic meta-heuristicslike evolutionary computation, simulated annealing, tabu search, neural computation, and so on, which are built around some basic principles taken by the observation of a particular natural phenomenon. As it is very common in the practical usage of these heuristics, ACO algorithms take often some distance from their inspiring natural metaphor. Often ACO algorithms are enriched with capacities that do not find a counterpart in real ants, like local search and global-knowledge-based actions, so that they can compete with more application-specific approaches. ACO algorithms so enriched are very competitive and in some applications they have reached world-class performance. For example, on structured quadratic assignment problems AS-QAP, HAS-QAP, and MMAS-QAP are currently the best available heuristics. Other very successful applications are those to the sequential ordering problem, for which HAS-SOP is by far the best available heuristic, and to data network routing, where AntNet resulted to be superior to a whole set of state-of-the-art algorithms.

Within the artificial life field, ant algorithms represent one of the most successful applications of swarm intelligence. One of the most characterizing aspect of swarm intelligent algorithms, shared by ACO algorithms, is the use of the stigmergetic model of communication. There are however examples of applications of stigmergy based on social insects behaviors other than ants foraging behavior. For example, the stigmergy mediated allocation of work in ant colonies has inspired models of task allocation in a distributed mail retrieval system, dead body aggregation and brood sorting, again in ant colonies, have inspired a data clustering algorithm, and models of collective transport by ants have inspired transport control strategies for groups of robots.

In conclusion, I think this paper has achieved its goal: to convince the reader that ACO, and more generally the stigmergetic model of communication, are worth further research.

6 References

[1] Marco Dorigo and Gianni Di Caro. Ant Algorithms for Discrete Optimization.
Universite Libre de Bruxelles Brussels, Belgium

[2] M. Dorigo and L. M. Gambardella. Ant colonies for the traveling salesman problem. BioSystems, 43:73{81, 1997.

[3] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics{Part
B, 26(1):29{41, 1996.

[4] E. Bonabeau, M. Dorigo, and G. Th_eraulaz. From Natural to Artificial Swarm
Intelligence. Oxford University Press, 1999.

[5] G. Reinelt. The Traveling Salesman Problem: Computational Solutions for TSP
Applications. Berlin: Springer-Verlag, 1994.